VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky

BAKALÁŘSKÁ PRÁCE

2016 **Petr Dobeš**

VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2016 **Petr Dobeš**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Petr Dobeš**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: KVM Quarda s.r.o.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Peter Chovanec**

Konzultant bakalářské práce: jaroslav Quarda

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě, dne 21. Dubna 2016 .....

Prohlášení zástupce spolupracující právnické osoby

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských/magisterských programech VŠB-TU Ostrava.

V Mošnově, dne 21. Dubna 2016 .....

Abstrakt

Obsahem této bakalářské práce je popis mého působení ve firmě KVM Quarda s.r.o. v rámci odborné praxe. Popsány jsou vybrané řešené úlohy. V závěru je zhodnocen přínos praxe a uplatněné znalosti získané během studia.

Klíčová slova: Praxe, Programování, PLC, Automatizace

Abstract

This bachelor work contains description of my professional practice in KVM Quarda s.r.o. Selected tasks are described. In the end there are evaluated profits and skills gained during studies that were used.

Keywords: Practice, Programming, PLC, Automation

Obsah

Seznam použitých zkratek a symbolů:

PLC – Programable logic controler

GUI – Graphical user interface

CAD – Computer-aided design

HMI – Human-machine interface

ADC – Analog-to-digital converter

XML – Extensible markup language

CSV – Comma-separated values

Seznam ilustrací a ukázek kódu

1 Úvod

Firma KVM Quarda s.r.o[1] v podobě, v jaké je známa nyní, funguje od roku 2014. Klíčovou roli zaujímají bratři Jaroslav a Radek Quardovi. Firma sídlí v Mošnově a zabývá se především řešeními z oblasti automatizace průmyslové výroby. Zákazníky jsou zejména výrobci z oblasti automotive. Typickým výstupem práce firmy jsou tedy jednoúčelová zařízení a přípravky sloužící k montáži či testování výrobků, například v automobilovém průmyslu. Firma má trvale 4 zaměstnance. Část práce je prováděna externisty, popřípadě v kooperaci s dalšími firmami.

Obr. 1: Logo firmy KVMQ s.r.o.

V roce 2016 probíhá rekonstrukce nově zakoupeného objektu, který bude sloužit pro rozšíření provozních prostor. Mezi zákazníky firmy patří například Mobis Automotive Czech s.r.o.[2], Varroc Lighting Systems s.r.o.[3] anebo Continental Automotive Czech Republic s.r.o.[4]

1.1 Struktura týmu

Výroby jednoúčelového zařízení se typicky účastní několik lidí různého profesního zaměření. Úkolem konstruktéra je navrhnout zařízení po stránce principu jeho funkčnosti a mechanického řešení. Jde tedy například o vytvoření 3D modelu celého zařízení pomocí CAD software. Elektroprojektant řeší návrh elektrického zapojení a podílí se na výběru elektronických komponent zařízení. Programátor pak má za úkol zařízení oživit.

To má zpravidla několik částí. První je seznámení se s konceptem zařízení a konzultace jak s konstruktérem, tak s elektroprojektantem. Následuje implementace programu a testování dílčích částí, pokud je to možné. Časově nejnáročnější je pak odladění programu na již sestrojeném a zapojeném zařízení. To nezřídka zahrnuje i pozdější návštěvy zákazníků z důvodu úprav a servisních zásahů.

2 Vybrané řešené úlohy

Během odborné praxe jsem se zabýval programováním řídicích systémů jednoúčelových zařízení. Šlo především o programovatelné logické automaty firmy Siemens. Součástí praxe také byla tvorba software pro PC, která s těmito zařízeními spolupracovala. Pracoval jsem tak především v prostředí Siemens TIA Portal[5] a Microsoft Visual Studio[6].

Při výkonu praxe jsem v rámci firmy spolupracoval především s konstruktérem zařízení a elektroprojektantem. Při oživování a předávání zařízení u zákazníka pak šlo o kooperaci s projektovými vedoucími, seřizovači a techniky různého zaměření.

2.1 Zatlačování klipů světlometů

Mým prvním projektem realizovaným během praxe bylo řešení řízení zařízení určeného pro zatlačování plechových klipů do plastových těl světlometů.

2.1.1 Výběr komponent

Po seznámení s konceptem zařízení a jeho plánovanou funkčností bylo nejprve třeba vybrat vhodný řídicí systém. Požadavkem zákazníka bylo použití PLC značky Siemens z důvodu unifikace náhradních komponent zařízení. Jelikož šlo o zařízení s nízkým počtem periferií, vybíral jsem z nejnižší řady PLC firmy Siemens. Po konzultaci s elektroprojektantem jsem zvolil typ S7-1214[7]. Za hlavní výhodu této volby považuji cenovou dostupnost všech základních CPU řady S7-1200. V tomto konkrétním případě navíc dostačuje počet již integrovaných digitálních vstupů/výstupů pro všechny periferie zařízení. V případě potřeby je navíc možno základní modul rozšířit o další vstupně-výstupní moduly.

2.1.2 Oživení zařízení

V době, kdy probíhaly na zařízení mechanické a elektrikářské práce, jsem se seznamoval se samotným způsobem programování programovatelných automatů. To spočívalo především ve studiu technických dokumentací a postupné implementaci programu zařízení. Za nejtěžší považuji především přechod na poněkud odlišný způsob uvažování nad strukturou programu. I přestože je program vykonáván sekvenčně, může představa paralelního zpracovávání jednotlivých programových bloků poněkud zjednodušit strukturu programu.

Během oživování stroje jsem narazil na různé drobné chyby jak v návrhu, tak i v samotné implementaci programu. Výsledkem bylo nicméně funkční zařízení.

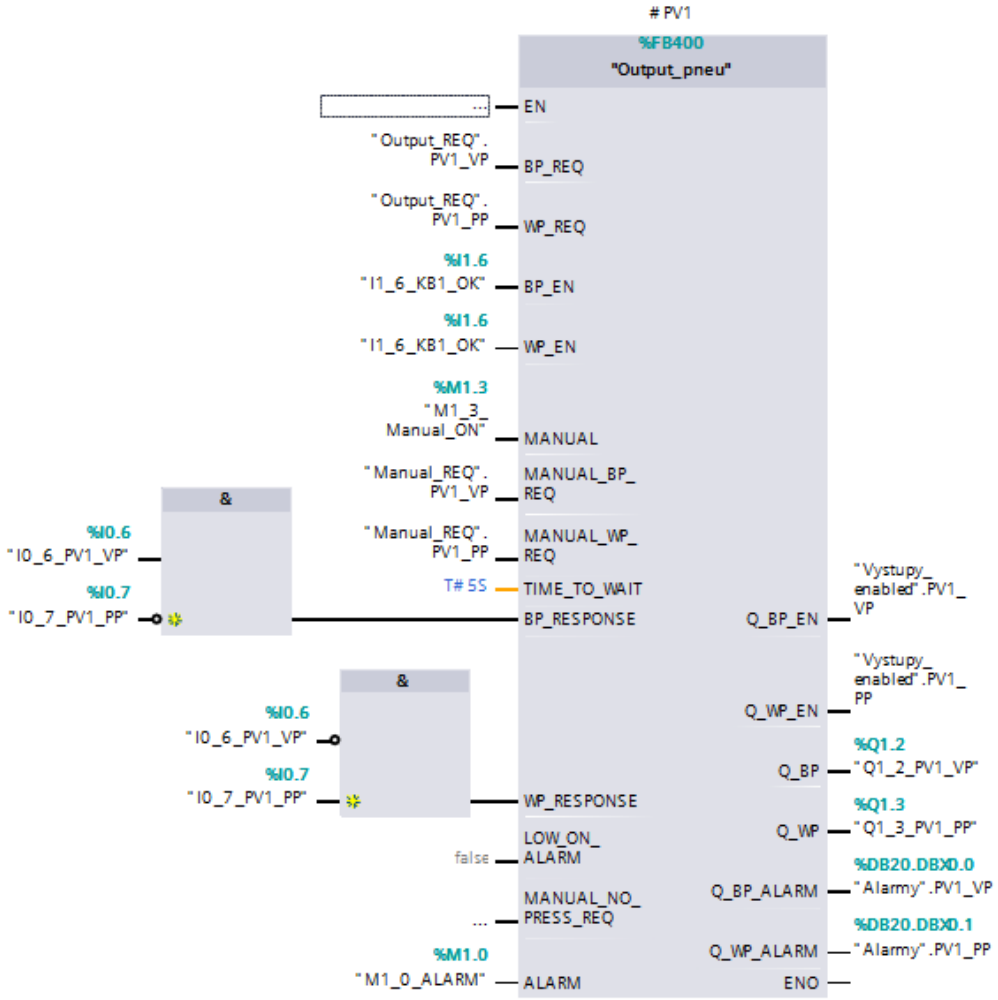
2.2 Testování pevnosti svarů světlometů

Jedním z dalších projektů bylo zařízení pro testování pevnosti svarů světlometů destruktivní zkouškou.

2.2.1 Organizace programu

Při návrhu tohoto projektu jsem se snažil využít toho, že některé úlohy se, jak jsem zjistil, na jednoúčelových zařízeních opakují, a bylo proto vhodné zjednodušit jejich řešení. Jde především o ovládání pneumatických prvků, podporu krokovaných sekvencí a podobně. Pro tyto problémy jsem si připravil podpůrné programové bloky, které zrychlují a zpřehledňují implementaci nadřazených programových bloků.

Například pro ovládání pneumatických válců nyní používám funkční blok „Output_pneu“. Pokud je nadřazený blok programován reprezentací logické sítě, což u bloků pro propojení komponent není neobvyklé, je grafická reprezentace znázorněna na obrázku č. 2.



Obr. 2: Programový blok pro obsluhu pneumatického válce

Vstupy BP_REQ a WP_REQ představují požadavky na přestavení pneumatického válce do výchozí anebo pracovní pozice (base position a work position). Vstupy BP_EN a WP_EN toto přestavení uvolňují tak, aby nemohla vzniknout kolize apod. TIME_TO_WAIT umožňuje nastavit maximální dobu pro přestavení válce, než bude nastaven některý z výstupů signalizujících poruchu. Vstupy označené prefixem MANUAL slouží pro ovládání zařízení v manuálním režimu typicky pomocí dotykové obrazovky umístěné na zařízení. Výstupy Q_BP a Q_WP už jsou skutečně přepisovány na fyzické výstupy (na obrázku Q1.2 a Q1.3). Využití tohoto bloku výrazně zrychluje a zpřehledňuje implementaci programu, což vede i k menšímu počtu chyb.

Samotná struktura programu je pak rozdělena do několika částí tak, aby byly další případné úpravy zařízení co možná nejsnadnější. To reflektuje také posloupnost implementace jednotlivých částí odspodu nahoru. Nejdříve jsem provedl hardwarovou konfiguraci zařízení. Poté jsem vytvořil nastavení zástupných symbolů pro fyzické adresy, což je nezbytné především pro přehlednost

rozsáhlejších programů. Jako první programovou část jsem pak implementoval ovládání výstupů zahrnující ochranu proti fyzickým kolizím ovládaných pohonů. To snižuje možnost poškození zařízení vlivem pozdějších chyb v programu. Dále je implementován blok pro řízení režimů zařízení a zpracování základních poruch. Souběžně tak může již být implementována vizualizace s manuálním ovládáním, kterého může využít například mechanik při mechanickém ladění zařízení. Automatický režim zařízení a jeho kompletní funkčnost jsem tak implementoval až jako poslední část na již vybudovaných základech. Tento postup se nakonec ukázal jako poměrně efektivní a využil jsem ho i v pozdějších projektech.

Obr. 3: Zařízení pro testování pevnosti svaru

2.2.2 Fungování zařízení

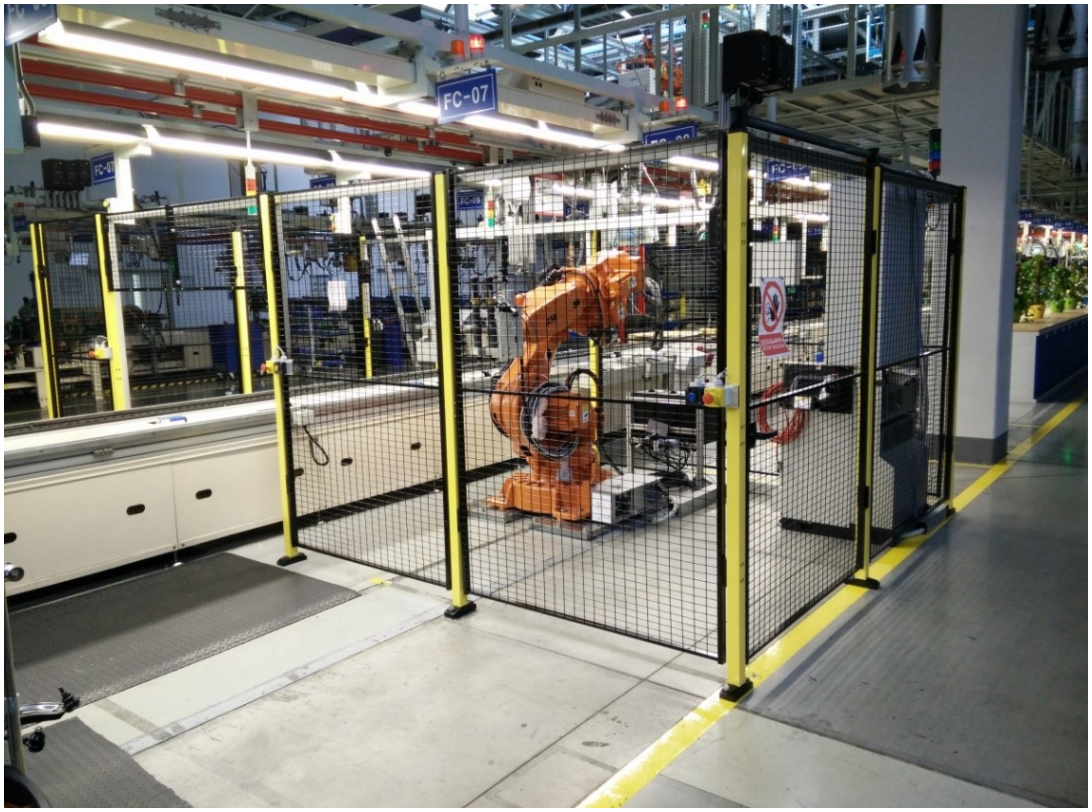
Pro měření síly utržení svaru je použit tenzometr. Analogový signál ze zesilovače je zapojen na jeden z integrovaných napěťových analogových vstupů. Ty měří v rozsahu od 0V do 10V ADC převodníkem s přesností 10 bitů. Získanou hodnotu je nutno přepočítat na jednotky síly. Zde se předpokládá, že napětí na výstupu zesilovače je lineárně závislé na zatížení tenzometru. Průběh hodnoty síly stejně tak jako její maximální hodnotu lze sledovat na dotykové obrazovce zařízení. Ta slouží i pro informování obsluhy o poruchách na zařízení a k nastavení parametrů stroje.

2.2.3 Pozdější úpravy zařízení

Dodatečným požadavkem zákazníka bylo propojení se zařízením, které světlomety svařuje, tak, aby v případě vyhodnocení špatného svaru byla výroba odstavena a nebyly dále vyráběny nevyhovující kusy.

2.3 Robotické stanoviště

Jednou ze složitějších řešených úloh bylo řízení robotického stanoviště pro utahování šroubů předních nápravnic automobilů. Řídicí systém měl za úkol obsluhovat komunikaci s ovladačem robota, komunikaci s výrobní linkou, řízení automatických utahovaček a ovládání servopohonu pro nastavení rozteče utahovaných šroubů.



Obr. 4: Robotické stanoviště pro utahování šroubů předních nápravnic

2.3.1 Komunikace s ovladačem robota

Pro komunikaci s ovladačem robota jsem zvolil často používanou sběrnici Profibus[8], která byla vyvinuta v Německu a je řídicími systémy Siemens často používána. Ta z pohledu programu simuluje běžné digitální vstupy a výstupy. Programátor je tak odstíněn od problémů navazování a udržování spojení. Zapotřebí je pouze správná konfigurace sběrnice. Na sběrnici se nachází jedno master zařízení (v tomto případě jde o řídící PLC) a dále pak řízená slave zařízení. S aplikačním technikem firmy ABB[20], jehož úkolem bylo naprogramování robota, bylo nejprve nutno dohodnout všechna předávaná data podle potřeb výrobního cyklu a komunikaci následně odladit.

2.3.2 Komunikace s výrobní linkou

Díky tomu, že v naprosté většině zařízení na výrobní lince zákazníka byly použity PLC značky Siemens, bylo možno využít stejného modelu komunikace jako na ostatních strojích jiných výrobců. Ten spočíval v poskytnutí vyhrazené oblasti paměti pro veřejný zápis i čtení pomocí protokolu S7. O čtení a zápis se pak cyklicky stará OPC[9] server zákazníka.

2.3.3 Řízení servopohonu

Mým úkolem bylo i správné nastavení ovladače servomotoru a jeho řízení. Motor zde byl použit pro pohon lineární osy upravující rozteč utahovaček. Ta se totiž pro různé typy automobilů liší. Komunikaci s ovladačem servomotoru bylo možno provést více způsoby. Nakonec byla využita nejjednodušší varianta a to pomocí digitálních vstupů a výstupů, kdy se vybírají již předem uložené akce v ovladači servomotoru. Komunikaci pomocí ethernetu v té době zvolený ovladač podporoval pouze v podobě proprietárního protokolu používaného softwarem pro nastavení ovladače. Novější verze tohoto ovladače však již podporují komunikaci např. přes protokol Modbus TCP[10], ke kterému výrobce dodává i knihovnu funkcí přímo pro PLC Siemens. Ta pak umožňuje téměř neomezený přístup k parametrům ovladače servopohonu. To by ovládání a úpravu nastavení ovladače učinilo značně pohodlnější a škálovatelnější.

2.3.4 Výběr řídicího systému

Pro řízení jsem nakonec vybral ověřené PLC S7-1214 s rozšiřujícím modulem pro sběrnici PROFIBUS a dvěma rozšiřujícími moduly s 16 digitálními vstupy a 16 digitálními výstupy. To pokrývá i rezervu pro případná rozšíření zařízení dalšími periferiemi bez nutnosti instalace dalších modulů.

2.3.5 Odladění zařízení

Protože jsme linku nemohli během provozu zastavit déle než na několik minut, bylo odladění za běhu poněkud komplikovanou částí řešení úlohy.

2.4 Software pro sledování změn parametrů výrobní linky

Požadavkem zákazníka bylo evidovat historii změn nastavení parametrů na jednotlivých zařízeních výrobní linky.

2.4.1 Volba použitých technologií

Jako perzistentní úložiště dat jsem zvolil databázi SQLite[11]. Důvodem je především jednoduchost implementace nevyžadující běh databázového serveru a z pohledu zákazníka jednoduché zálohování dat realizovatelné pouze pomocí běžného kopírování souboru databáze.

Nastavení aplikace obsahující seznam zařízení a jejich sledované parametry je uloženo v souboru ve formátu XML. Ten je snadno parsovatelný a v případě potřeby i ručně upravitelný. Ukázku nastavení představuje obrázek č. 5. V nastavení je uloženo id zařízení, jeho IP adresa, data potřebná pro připojení a seznam všech sledovaných parametrů včetně názvu, datového typu a adres datových bloků potřebných k jeho stažení.

Jako programovací jazyk jsem zvolil jazyk C# spolu s frameworkem .NET[12]. Cílovým operačním systémem byl Microsoft Windows 7.

```
<machine>
  <id>2</id>
  <Name>Test machine 3</Name>
  <IP>192.168.10.100</IP>
  <ConnectionString>rack:0;slot:1</ConnectionString>
  <PLCType>S7 PLC</PLCType>
  <parameters>
    <parameter>
      <Name>Force 1</Name>
      <DataType>REAL</DataType>
      <DBNumber>50</DBNumber>
      <DBOffset>20</DBOffset>
    </parameter>
    <parameter>
      <Name>Force 1 - Tol plus</Name>
      <DataType>REAL</DataType>
      <DBNumber>50</DBNumber>
      <DBOffset>24</DBOffset>
    </parameter>
    ...
  </parameters>
</machine>
```

Obr. 5: Ukázka nastavení v XML

2.4.2 Architektura

Architekturu aplikace jsem rozdělil do několika částí.

2.4.2.1 Model

Jako první jsem navrhnul vrstvu představující model skutečnosti, tedy jednotlivá zařízení a seznam sledovaných parametrů a jejich typy. Model zde spočíval spíše ve struktuře entit než jejich chování. Částí modelu však bylo i řešení samotného připojení a stažení dat ze zařízení. To je prozatím implementováno pouze pro PLC firmy Siemens řady S7, protože ta jsou výlučně u zákazníka využívána jako standard a jsou obsazena na všech zařízeních linky. Návrh aplikace však s rozšířením o jiné typy PLC počítá.

Protože model je navržen velmi jednoduše, použil jsem návrhový vzor active record. Datová vrstva tak není od modelu oddělena.

2.4.2.2 Servisní vrstva

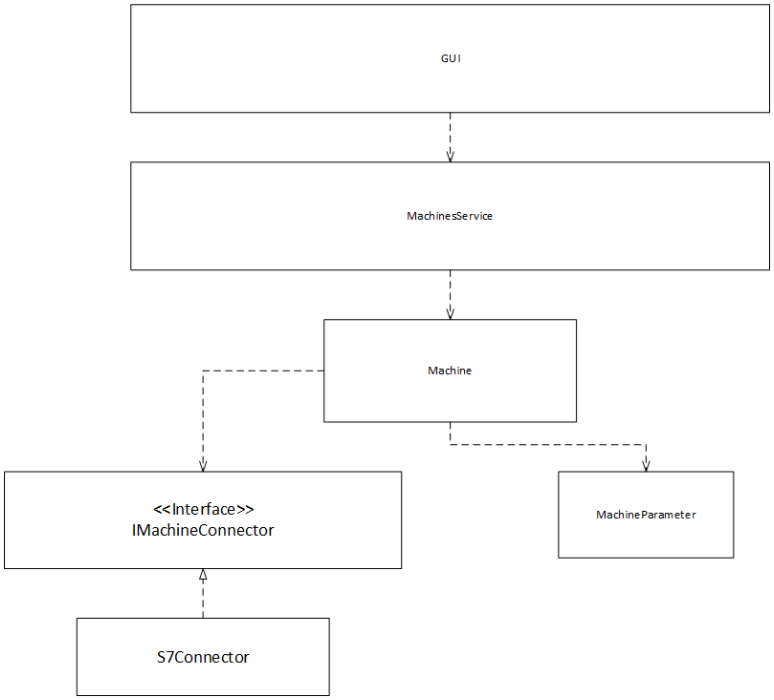
Další vrstvou je vrstva servisní, jejímž úkolem je vytvoření instancí jednotlivých tříd modelu a jejich obsluha. Stará se tak například o kontrolu změn parametrů. Ta probíhá v samostatném vlákně, které udržuje připojení ke všem zařízením a cyklicky stahuje všechny sledované parametry.

Aby nebylo potřeba vždy kontrolovat změnu všech jednotlivých parametrů, jsou všechny hodnoty převedeny na jeden textový řetězec, ze kterého je pak vypočítána hash, která se porovnává s poslední známou hash z databáze. V případě rozdílnosti hash hodnot je seznam hodnot uložen jako serializovaný lob do databáze. V lobu není uložena pouze hodnota, ale i název a typ sledovaného parametru. To sice zavádí redundanci dat, za to však je uchován i seznam sledovaných parametrů přímo s jejich hodnotami a není tak zapotřebí dalších tabulek k ukládání historie sledovaných parametrů.

2.4.2.3 Vrstva uživatelského rozhraní

Vrstva uživatelského rozhraní je nejvyšší vrstvou. Umožňuje sledovat historii nastavení na jednotlivých zařízeních, měnit seznam zařízení či sledované parametry.

Reprezentaci architektury lze vidět na obrázku č. 6.



Obr. 6: Zjednodušené schéma architektury

2.4.3 Implementace

Klíčovou otázkou předcházející samotnou implementaci byl výběr vhodného způsobu získávání dat z PLC.

Jednou z možností bylo využití vlastního protokolu pomocí TCP/IP. To by však vyžadovalo netriviální a především časově náročnou úpravu softwaru na všech stávajících zařízeních. I rozšiřitelnost o nová zařízení by byla zkomplikována. Vzhledem k tomu, že jsou na výrobní lince použita pouze PLC firmy Siemens, jevilo se nejvýhodnější využít protokol S7, který tato PLC implementují implicitně. Volba se tedy zúžila pouze na výběr vhodné knihovny, protože implementovat protokol vlastními silami by bylo časově zbytečně náročné vzhledem k dostupným řešením.

Testovanými knihovnami byly knihovny Libnodave[13] a Snap7[14]. Obě nabízejí podobná rozhraní pro jejich využití. Srovnání minimálního množství kódu pro vyčtení jednoho bytu dat z datového bloku je možno vidět na obrázku č. 7 a obrázku č. 8.

```
byte data;
libnodave.daveOSserialType daveSerialType;
libnodave.daveInterface daveInterface;
libnodave.daveConnection daveConneciton;

daveSerialType.rfd = libnodave.openSocket(102, "192.168.10.10");
daveSerialType.wfd = daveSerialType.rfd;
if (fds.rfd > 0)
{
    daveInterface = new libnodave.daveInterface(daveSerialType, "IF1", 0, libnodave.daveProtoISOTCP, libnodave.daveSpeed500k);
    daveInterface.setTimeout(10000);
    daveConneciton = new libnodave.daveConnection(daveInterface, 0, rack, slot);

    if (daveConneciton.connectPLC() == 0)
    {
        if (daveConneciton.readBytes(libnodave.daveDB, 10, 0, 1, null) == 0)
        {
            data = (byte)daveConneciton.getS8();
            dc.disconnectPLC();
        }
    }
}
libnodave.closeSocket(fds.rfd);
```

Obr. 7: Minimální kód pro připojení pomocí knihovny Libnodave

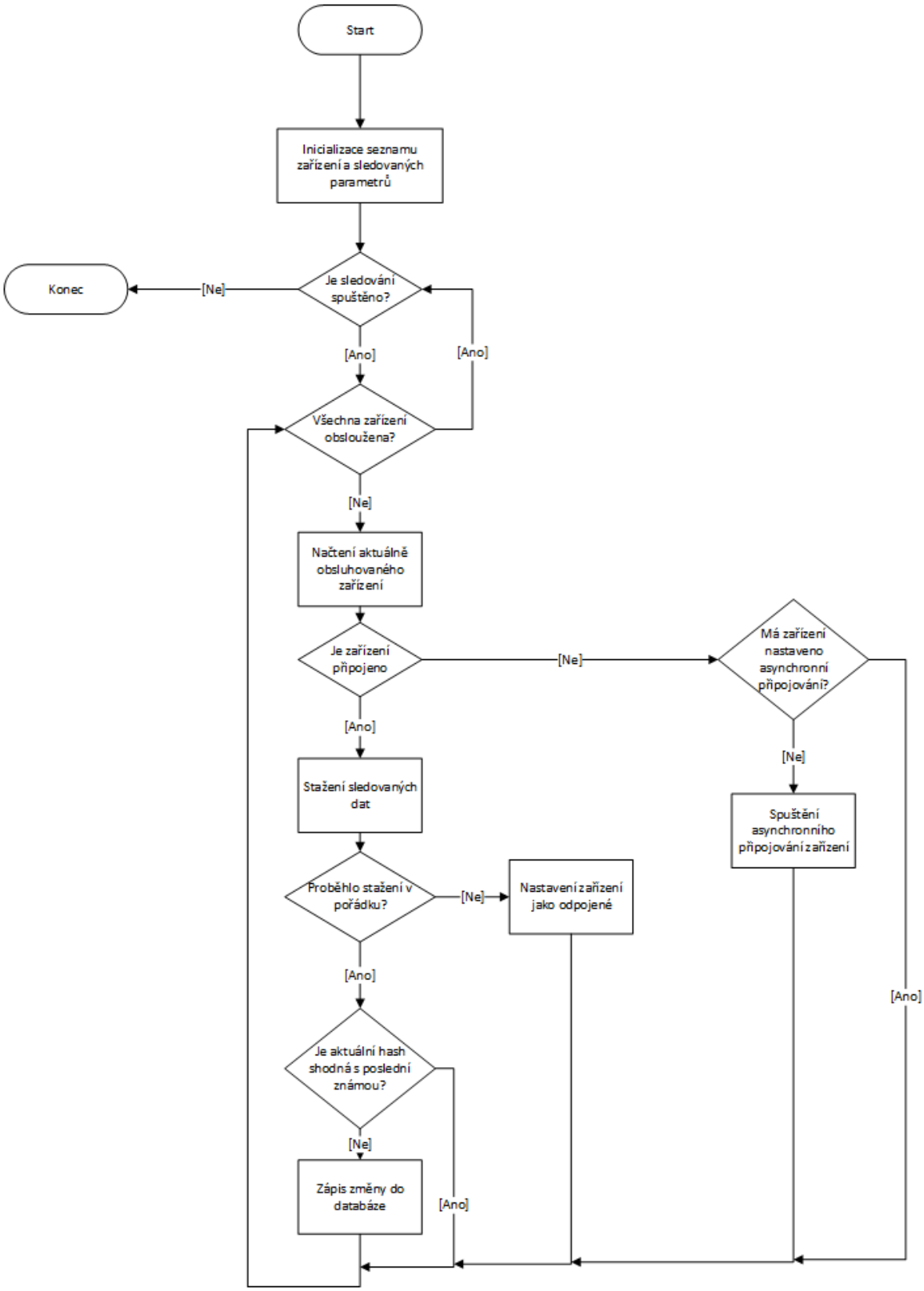
```
byte data;
S7Client client = new S7Client();
if (client.ConnectTo("192.168.10.10", 0, 1) == 0)
{
    byte[] Buffer = new byte[0x10000];
    int BufferSize = Buffer.Length;
    client.DBRead(10, 0, 1, Buffer);
    data = Buffer[0];
}
```

Obr. 8: Minimální kód pro připojení pomocní knihovny Snap7

Během testovaní s PLC však u knihovny Libnodave ze zatím nezjištěné příčiny několikrát došlo k zápisu chybné hodnoty. Knihovna Snap7 fungovala bezchybně a i její výkonnost se jevila dostačující. Volba tedy padla na ni.

Při testování aplikace se ukázalo, že pokud je v seznamu zařízení větší počet zařízení, která nejsou skutečně připojena (jsou například vypnuta), dochází k výraznému zpomalení aplikace při pokusech o připojení. Jedno odpojené zařízení tak způsobovalo zpoždění v řádu i jednotek vteřin během každého cyklu. Bylo proto třeba drobně upravit mechanismus obsluhování zařízení a připojování. Jde především o zavedení asynchronního připojování k odpojeným zařízením. Pokud zařízení není označeno jako připojené, je pro něj spuštěno nové vlákno řešící připojování, které obsluhu ostatních zařízení nezpomaluje.

Na vývojovém diagramu, znázorněném na obrázku č. 9 je možno vidět fungování samostatného vlákna ve třídě MachinesService, které se stará o cyklické stahování dat ze zařízení.



Obr. 9: Vývojový diagram znázorňující připojení a zpracování dat

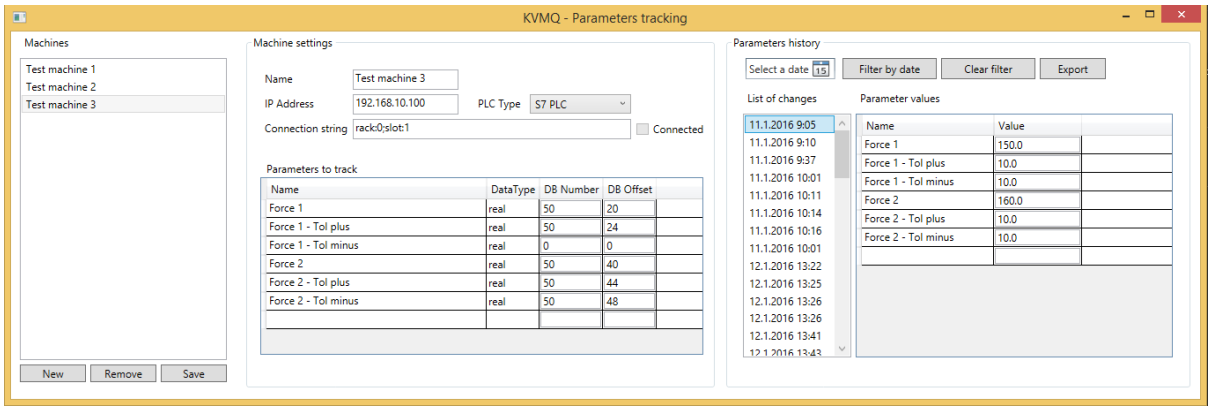
2.4.4 GUI

Na obrázku č. 10 lze vidět uživatelské rozhraní programu. To je rozděleno do tří částí.

Levá část slouží k manipulaci se seznamem sledovaných zařízení.

Po vybrání zařízení z levé části je možno editovat nastavení zařízení v prostřední části okna. Ta obsahuje také seznam sledovaných parametrů. U každého sledovaného parametru je evidován jeho název, datový typ a číslo datového bloku, ze kterého je načítán a adresa uvnitř datového bloku.

V pravé části okna lze pak u vybraného zařízení sledovat historii nastavených parametrů. Tu je pak možno filtrovat podle dne změny nastavení. Zobrazená data je pak možno exportovat ve formátu CSV.

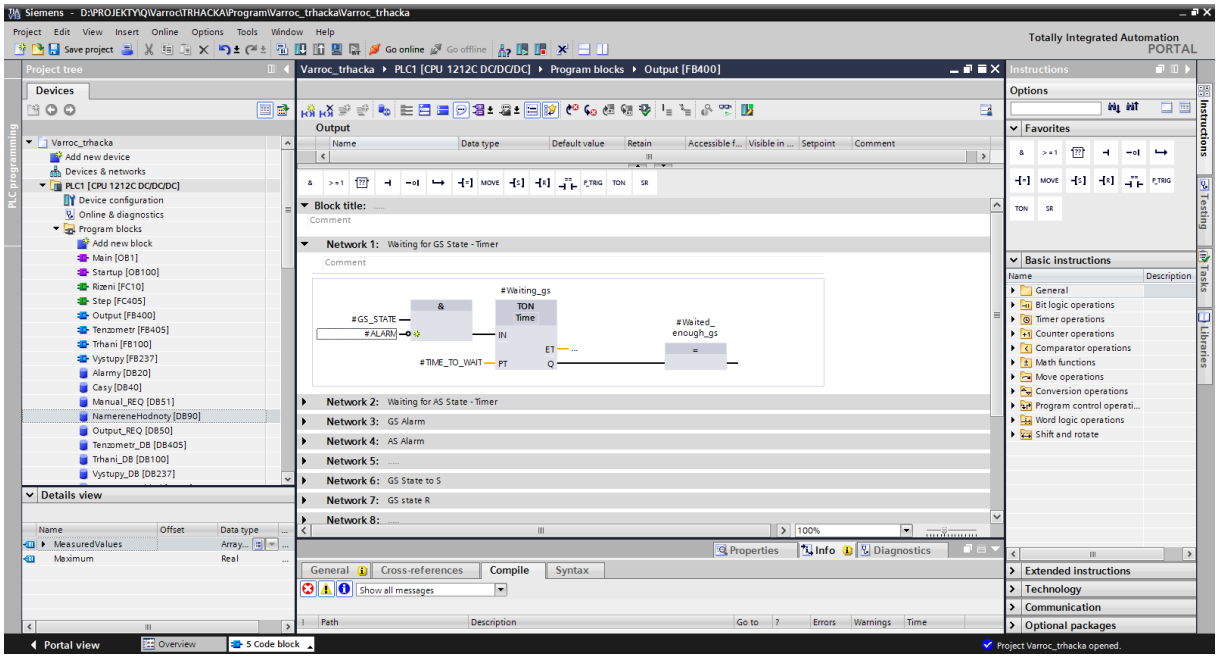


Obr. 10: Uživatelské rozhraní

3 Použité technologie

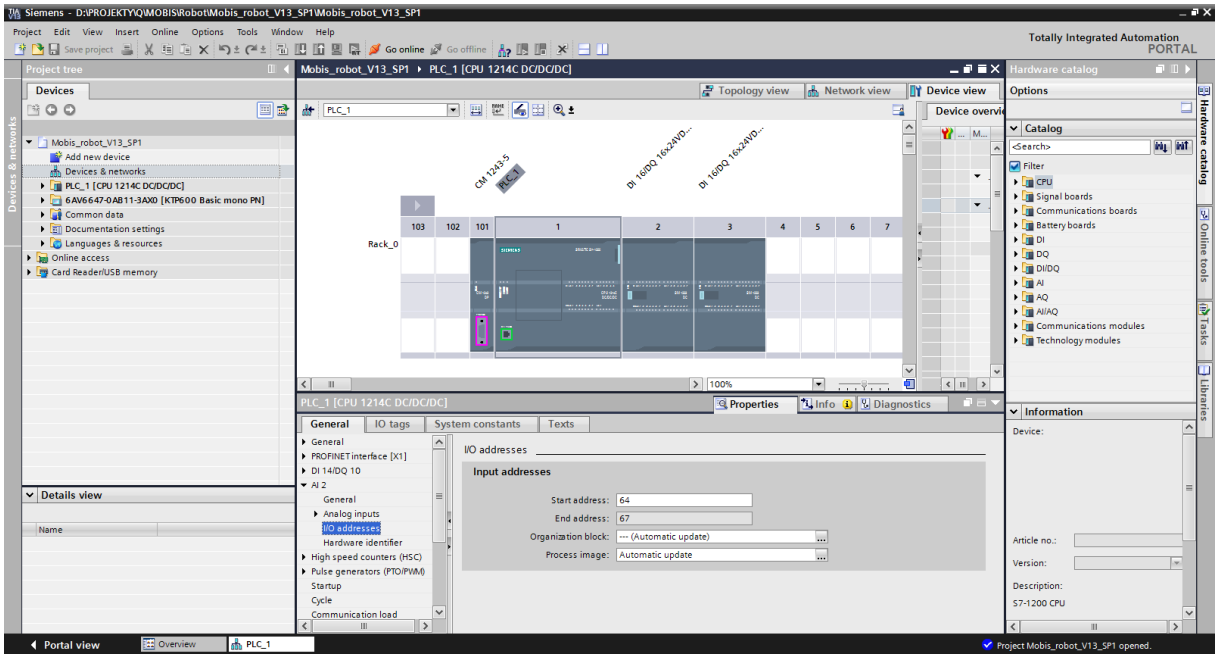
3.1 Softwarové nástroje

Jelikož velkou část náplně mojí práce tvořilo programování PLC značky Siemens, pracoval jsem s prostředím Siemens TIA Portal V13. To je následovníkem dodnes používaného Simatic Step 7[15]. Kromě toho, že podporuje i poslední verze PLC, integruje i možnost vytváření vizualizací pro dotykové panely HMI[16]. Tím jsem byl upoután výhradně na použití operačního systému Microsoft Windows, protože Siemens TIA Portal pod jinými OS není možno provozovat.



Obr. 11: Prostředí TIA Portal - Funkční blok

Na obrázcích č. 11 a č. 12 je možno vidět prostředí Siemens TIA Portal. V levé části je seznam všech řídicích systémů, HMI a dalších periferií daného projektu. Pro každý programovatelný prvek je pak zobrazen i seznam všech programových bloků (to zahrnuje jak datové bloky, tak i funkční bloky) a nástrojů pro konfiguraci či diagnostiku. Ve střední části je pracovní okno zobrazující aktuálně otevřenou položku. Může jít například přímo o nějaký programový blok či hardwarovou konfiguraci. Ta je vidět na obrázku č. 12. V pravé části okna je pak seznam aktuálně použitelných nástrojů, což v případě funkčních bloků zahrnuje kompletní instrukční sadu a načtené knihovny.



Obr. 12: Prostředí TIA Portal - Hardwarová konfigurace

V několika projektech bylo třeba vyřešit sběr dat z PLC pomocí PC. Pro tyto účely jsem zvolil knihovnu Snap7, která zastřešuje proprietární protokol S7. Jako primární jazyk jsem pak volil C# s frameworkem .NET. Pro psaní kódu, jsem pak využíval vývojového prostředí Microsoft Visual Studio.

3.2 Hardwarové nástroje

Pro ovládání zařízení byly nejčastěji použity PLC Siemens řady S7[17]. Ta byla představena roku 1995 jako náhrada za stárnoucí řadu S5[18]. Řídicí systémy řady S7 jsou používány ve velkém počtu automatizovaných výrob zejména v Evropě a jsou zákazníky často vyžadovány.

Organizace programu pro tato PLC se zpravidla dělí do tzv. bloků různého typu. Datové bloky jsou paměťové oblasti sloužící čistě pro uchování dat a lze jim přidělit různé vlastnosti. Např. zda data přetrvávají i po odpojení napájení v paměti vyhrazené pro trvalé uchování dat, ta je ve většině PLC realizována pamětí typu flash. Funkční bloky naopak popisují chování a lze je psát několika způsoby. Jedním ze způsobů je grafické znázornění logickou sítí. Pro řešení jednoduchých problémů jde o přehlednou a rychlou variantu. Pro složitější problémy lze využít strukturovaného jazyka (SCL[19]), který lze syntaxí přirovnat k jazyku Pascal.

4 Znalosti získané v průběhu studia uplatněné při výkonu praxe

Během praxe jsem využil znalostí z oblasti logických obvodů a elektrotechniky získaných jak studiem na střední, tak i vysoké škole.

Jako velmi přínosná se ukázala znalost programování v jazyce C# v kombinaci s frameworkem .NET osvojená především absolvováním předmětů Programovací jazyky II a Architektura technologie .NET.

Užitečné při výkonu praxe se ukázaly i znalosti z oblasti počítačových sítí.

Protože v zákaznických firmách často pracují i lidé ze zahraničí, využil jsem také znalosti anglického jazyka.

5 Znalosti chybějící při výkonu praxe

Vzhledem k tomu, že částí náplně mé praxe byly i úkoly vyžadující znalosti z jiného než studovaného oboru, musel jsem tyto znalosti získat samostudiem. Jedná se především o problematiku senzoriky a řídicích systémů obecně. Dozvěděl jsem se také mnoho o elektrických a pneumatických pohonech a jejich řízení.

Protože jsem komunikoval i s různě odborně zaměřenými zástupci zákaznických firem, ocenil bych větší zkušenosti z oblasti soft-skills.

6 Zhodnocení praxe

Za velký přínos praxe považuji především získání celkového povědomí o fungování vývoje zařízení v oblasti automatizace, kterou nyní i považuji za svůj cílový obor.

Během praxe jsem získal znalosti programování PLC. Ta jsou nyní v podstatě nutností v každé automatizované výrobě, což reflektuje i poptávka na trhu práce. Odborná praxe tak zvýšila moji šanci na budoucí pracovní uplatnění.

Velmi zajímavá byla i možnost spolupráce se zkušenějšími techniky z různých oborů a získání povědomí o aktuálně používaných technologiích.

Reference

[1] KVMQ s.r.o. URL: <http://www.kvmq.cz/> [cit. 2016-04-19]

[2] Mobis automotive Czech s.r.o. URL: <http://www.mobis-auto.cz/> [cit. 2016-04-19]

[3] Varroc Lighting Systems s.r.o. URL: < http://www.varroc.cz/o-nas-2.html> [cit. 2016-04-19]

[4] Continental Automotive Czech Republic s.r.o. URL: <http://www.continental-corporation.com/www/hr_cz_cz/themes/ov1_locations_cz/ov1_frenstat_pod_radhostem_cz/> [cit. 2016-04-19]

[5] TIA Portal. URL: <http://www.industry.siemens.com/topics/global/en/tia-portal/pages/default.aspx> [cit. 2016-04-16]

[6] Visual Studio – Microsoft developer tools. URL: <https://www.visualstudio.com/> [cit. 2016-04-16]

[7] S7-1214 - 6ES7214-1AG40-0XB0 - Product Details. URL: <https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6ES7214-1AG40-0XB0> [cit. 2016-04-16]

[8] Profibus – Overview. URL: <http://www.profibus.com/technology/profibus/overview/> [cit. 2016-04-16]

[9] Data Communication between S7 Station and PC Station, using SIMATIC NET OPC Server. URL: <https://support.industry.siemens.com/cs/document/67295801/data-communication-between-s7-station-and-pc-station-using-simatic-net-opc-server?dti=0&lc=en-CA> [cit. 2016-04-16]

[10] The modbus organisation. URL: <http://www.modbus.org/> [cit. 2016-04-16]

[11] SQLite. URL: <http://www.sqlite.org/> [cit. 2016-04-16]

[12] Microsoft .NET. URL: <https://www.microsoft.com/net/default.aspx> [cit. 2016-04-16]

[13] Libnodave. URL: <https://sourceforge.net/projects/libnodave/> [cit. 2016-04-16]

[14] Snap7. URL: <http://snap7.sourceforge.net/> [cit. 2016-04-16]

[15] Simatic Step 7. URL: <http://w3.siemens.com/mcms/simatic-controller-software/en/step7/step7-professional/pages/default.aspx> [cit. 2016-04-16]

[16] Siemens HMI. URL: <http://w3.siemens.com/mcms/automation/en/human-machine-interface/pages/default.aspx> [cit. 2016-04-16]

[17] Řada S7. URL: <http://w3.siemens.com/mcms/programmable-logic-controller/en/advanced-controller/s7-300/pages/default.aspx> [cit. 2016-04-16]

[18] Řada S5. URL: <http://w3.siemens.com/mcms/process-control-systems/en/simatic-pcs7-migration/simatic-s5/pages/simatic-s5.aspx#Strategy> [cit. 2016-04-16]

[19] Jazyk SCL. URL: <http://w3.siemens.com/mcms/simatic-controller-software/en/step7/simatic-s7-scl/pages/default.aspx> [cit. 2016-04-16]

[20] ABB Robotics. URL: < http://new.abb.com/products/robotics> [cit. 2016-04-19]